

# Analyzing the Structure of a Motion Capture Database under a Similarity Metric

Donovan H. Parks

Electrical and Computer Engineering, UBC, Vancouver, Canada



**Abstract**—The film and video game industries make use of large motion capture databases for creating realistic animations of human motion. Recently, significant research efforts have been devoted to data-driven animation techniques. These techniques allow new, realistic motion sequences to be automatically synthesized from existing motion capture sequences that are *similar* to the desired motion sequence. Unfortunately, identifying similar motion sequences is a challenging problem since *logically* similar motion sequences are often not *numerically* similar.

This paper introduces an interactive visualization environment for analyzing the structure that is imposed on a human motion database by a given similarity metric. Our environment provides insights into similarity metrics that are difficult to obtain by analyzing numerical results or by using existing visualization environments. We illustrate the use of our visualization environment by analyzing a recently developed similarity metric which is applied to a subset of the CMU motion capture database.

**Index Terms**—visualization, motion capture, similarity metric, proximity data, multidimensional scaling

## 1 INTRODUCTION

Large motion capture (mocap) databases are commonplace in the film and video game industries, where they are used to achieve realistic animations of human motion. Recently, significant research efforts have been devoted to developing methods for identifying similar motion sequences in order to allow animators to quickly find similar motions within a mocap database [1]–[3] or to synthesize new, realistic motions by morphing and blending existing mocap data [4], [5]. Identifying similar motion sequences is a challenging problem since *logically* similar motion sequences are often not *numerically* similar. For example, motion sequences of individuals boxing are logically similar although they differ significantly if numerically compared on a frame-by-frame basis. Thus, the essential problem is to define a similarity metric that can “bridge the semantic gap between logical similarity as perceived by humans and computable numerical similarity” [3].

Numerous similarity metrics have been defined for comparing mocap sequences (*e.g.*, [2], [3], [6]–[8]). Which of these should be preferred? What are their respective strengths and weaknesses? How can a given metric be improved? Is the structure imposed on a mocap database by a given similarity

metric useful for indexing the database or for the automatic synthesis of new motions?

This paper introduces an interactive visualization environment that can aid in answering questions of this nature. Our environment makes use of established InfoVis practices (*i.e.*, multidimensional scaling, coordinated views, details-on-demand, and visual encoding principles) which facilitate the rapid understanding of the structure imposed on a mocap database by a given similarity metric. This permits key aspects of a similarity metric to be quickly understood. Our environment provides insights into similarity metrics that are difficult to obtain by using existing visualization environments or by analyzing numerical results. To illustrate the use of this visualization environment, the similarity metric proposed by Li and Prabhakaran [7] is analyzed on a subset of the CMU mocap database<sup>1</sup>.

Figure 1 gives an overview of our proposed MotionVis environment. Given a similarity metric of interest, a dissimilarity matrix is constructed which indicates the dissimilarity between all pairs of mocap sequences in a mocap database. Multidimensional scaling (MDS) is then applied to the dissimilarity matrix in order to allow its structure to be visualized using a 2-dimensional scatterplot. By tightly coupling this scatterplot view with auxiliary *details-on-demand* views critical information about the similarity metric can be quickly obtained.

The direct contribution of this paper is the design and evaluation of an interactive visualization environment for analyzing mocap-based similarity metrics. We justify our design choices and demonstrate that our system provides valuable insight into the strengths and weaknesses of a mocap-based similarity metric. The broader contribution of this paper is demonstrating, by means of example, that a well-designed visualization environment can provide valuable insight into a similarity metrics. Similarity metrics abound in the natural and applied sciences and a general framework for gaining insight into their operation would be of great value. The application of visualization techniques to similarity metric understanding, as opposed to traditional data understating, is an important and relatively unexplored area of research that we believe is worth the attention of the InfoVis community.

The remainder of this paper is organized as follows. In Section 2 we describe related work. An overview of our visualization environment is given in Section 3. In Section 4

*e-mail:* donovanp@ece.ubc.ca

1. mocap.cs.cmu.edu

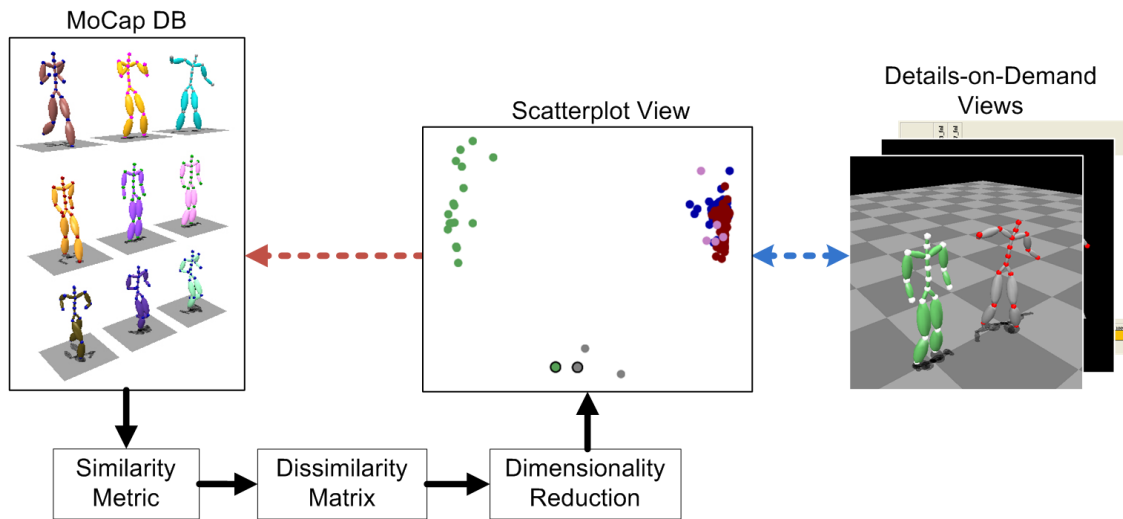


Fig. 1. MotionVis is an interactive environment for analyzing the structure that is imposed on a human motion database by a similarity metric. Given a similarity metric of interest, a dissimilarity matrix can be constructed for the mocap database. MDS can then be applied to the dissimilarity matrix in order to allow its structure to be visualized using a scatterplot. Tight coupling between the scatterplot view and details-on-demand views allows critical information about the similarity metric to be quickly obtained.

we explore issues specific to using mocap data and in Section 5 we discuss methods for performing MDS. We formally justify our design decisions in Section 6 and illustrate the use of our visualization environment in Section 7. A discussion of the strengths, limitations, and potential future work - including generalizing our environment beyond mocap data - is given in Section 8.

## 2 RELATED WORK

This work is unique in that its aim is to provide a visualization environment for understanding a similarity metric. However, many of the individual elements of MotionVis are in mainstream use and have a rich history of research. The work related to MotionVis can be divided into three categories: motion visualization, high-dimensionality visualization, and visual encoding.

### 2.1 Motion visualization

Two different aspects of motion visualization are discussed in this section. We begin by considering the large number of software packages that have been developed for creating animations with mocap data. Although the goals of these packages are different than ours, they contain many useful conventions which we have adopted and extended. Next, we discuss two key studies that investigate how the human brain processes human motion. The results of these studies validate the use of simplified human models for visualizing mocap data.

Several commercial programs (*e.g.* Maya, Poser) and open source programs (*e.g.*, Blender, MotView, CMU mocap Player) exist for manipulating mocap data and visualizing it in the form of an animated 3-dimensional skeleton. These programs range from complete animation packages used extensively

by the film and movie industries to simple mocap viewing utilities used by hobby animators. They are not intended to address the problem of understanding a similarity metric. Nevertheless, we adopt and extend a number of conventions used by these programs. Most notably, MotionVis provides two details-on-demand views which allow mocap sequences to be visualized as an animated 3-dimensional skeleton. We also allow the user to switch between a number of preset views (*i.e.*, isometric, front, side, top) that have become ubiquitous in these programs. Like many of these programs, we also make use of a tiled ground plane and simple shadows to help establish the 3-dimensional position of an object within a virtual world (see [9] for an introduction to different depth cues). Finally, we modify the concept of end-effector tracing developed in MotView<sup>2</sup> to that of path tracing in order to allow the user to easily visualize the path traveled during a mocap sequence.



Fig. 2. Two frames from a point-light display generated from mocap data of a walking sequence [10].

Johansson [11] developed the *point-light display* in 1975

<sup>2</sup>. [www.cs.wisc.edu/graphics/Courses/cs-838-2000/Students/gardner/motView/](http://www.cs.wisc.edu/graphics/Courses/cs-838-2000/Students/gardner/motView/)

to investigate the capabilities of the human brain in the context of understanding human motion. In a point-light display, lights are attached to a person and their movements recorded in a darkened room so only the moving lights are visible (see Figure 2). It was established that test subjects watching such a video could immediately infer a walking human. This demonstrates that humans are adept at understanding human motion even from a very sparse amount of visual information. As such, it is reasonable for animation packages and our visualization environment to animate mocap sequences using a simplified model of the human body.

Hodgins *et al.* [12] conducted experiments with a simple stick figure model and a more detailed polygonal model to determine if a viewer’s perception of motion characteristics is affected by the geometric model used. Their results indicate that the perception of motion is influenced by the geometric model used to render the motion. For this reason, we allow the user to select between simple stick figure models and more complex cylinder and ellipsoid-based models.

## 2.2 High-dimensionality visualization

Many techniques have been developed for visualizing high-dimensional data sets. Although these techniques are generally concerned with extracting meaning from a dataset (as opposed to gaining insight into an algorithm operating on the dataset), this work has influenced the design of our system.

A critical design decision in our system is to use MDS in order to allow a dissimilarity matrix to be viewed as a simple scatterplot. As an alternative, we could have employed a scatterplot matrix (or more generally, a multiform matrix [13]) or parallel coordinates [14] in order to visualize all dimensions of the dissimilarity matrix. In particular, we could have made use of the XmdvTool [15] or XGobi [16] visualization environments which consist of a number of predefined visualizations (*e.g.*, scatterplot matrix, star glyphs, and parallel coordinates) for exploring high-dimensional data sets. Although the visualization techniques provided in these environments excel at providing insight into the relationships between dimensions, they are not as effective at indicating the relationship between pairs of data points. Since we are interested in understanding the similarity metric which defines the dissimilarity between pairs of data points we favour a visualization environment that focuses on this aspect of the data.

Techniques such as self-organizing maps (SOM) [17] and graph layout algorithms [18] can be used to achieve low dimensional visualizations of high dimensional data, but we prefer MDS since it directly aims to preserve the dissimilarity between pairs of data points [19]. do not directly aim to preserve pairwise dissimilarity. A notable visualization environment that is tailored to proximity data is XGvis [20]. XGvis is an interactive visualization environment that incorporates a number of MDS techniques along with useful pre- and post-processing techniques (*e.g.*, lower and upper trimming of dissimilarities, random removal of dissimilarities for stability checks). Unfortunately, being a general visualization tool it lacks data specific views. We feel data specific, details-on-demand views are essential for understanding the structure

imposed on a database by a similarity metric. This is especially true when the underlying data is as complex as a mocap sequence. As such, our visualization environment consists of a number of details-on-demand views which are tightly coupled with the scatterplot view obtained using MDS.

Arguably the work most similar to ours is that of Sakamoto *et al.* [1]. They have developed an interactive environment for navigating through a mocap database. They employ SOM to arrange pose data from mocap sequences onto a 2-dimensional grid. Similar poses are then clustered together and a *characteristic pose* is used to represent the cluster and its general location on the grid. The user can then search for mocap sequences by selecting characteristic poses that would appear in the sequence they are interested in. This work is similar to our own in the sense that it is concerned with visualization in the context of mocap data. However, their interest is in providing a visualization that focuses specifically on the mocap data, whereas our goal is to develop a visualization environment that will aid in the understanding of a mocap-based similarity metric.

## 2.3 Visual encoding

Proper visual encoding is essential in a visualization environment that aims for rapid exploration and understanding. The most critical encoding in our system is distinguishing between different motion classes in the scatterplot view. We follow Mackinlay’s [21] suggestion of encoding nominal data using hue. The exact colours selected follow the advice of Ware [9]. Berlin and Kay [22] conducted a study of more than 100 languages and determined that there is a relatively fixed order in which colour names appear in languages. Ware’s colour suggestions are based on the most common 11 colours found in the Berlin and Kay study. Mackinlay also indicates that shape is a suitable method for encoding nominal data. We make use of shapes for relating selected points in the scatterplot view to their skeletons in our details-on-demand views (see Figure 8).

## 3 OVERVIEW

Our MotionVis environment has been designed to support the interactive exploration of the structure imposed on a mocap database by a given similarity metric. The goal of this environment is to allow key aspects of a similarity metric to be quickly understood. The MotionVis interface is shown in Figure 3. It consists of a details-on-demand section (left) and a scatterplot section (right), along with a number of controls for interacting with these views. We provide three different details-on-demand views: the skeletal view, the motion map view, and the dissimilarity metric view.

### 3.1 Scatterplot view

The scatterplot view is a visualization of the structure imposed on the mocap database by a given similarity metric. Figure 1 illustrates how this scatterplot view is obtained. Given a similarity metric of interest, a dissimilarity matrix is constructed which indicates the dissimilarity between all pairs of

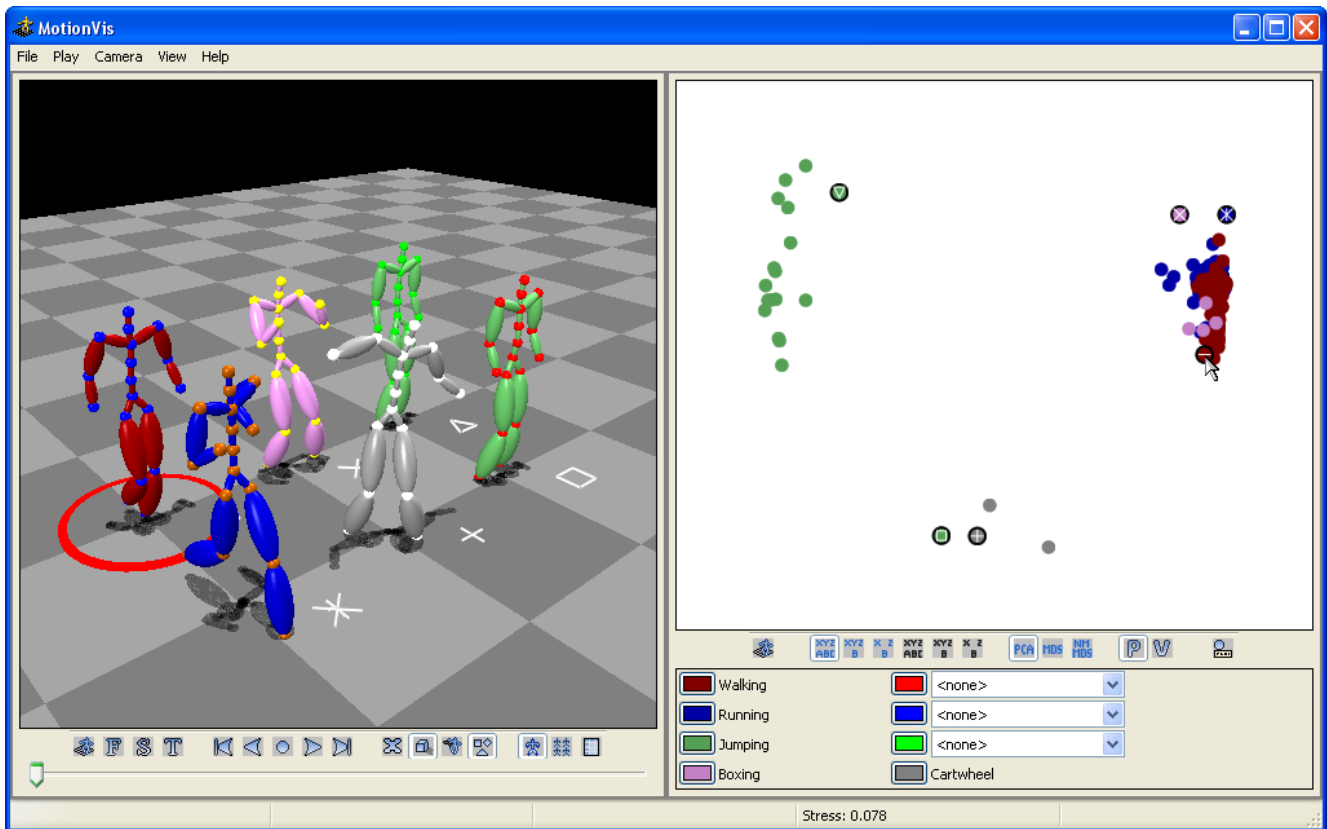


Fig. 3. The MotionVis interface is divided into two main sections: a details-on-demand section (left) and a scatterplot section (right). Different features of the program are made available through menu options and controls provided below each section. Here, the details-on-demand section is set to the skeletal view which allows for the direct visualization of the mocap sequences associated with the selected points in the scatterplot view (shown by black circles). Shapes are used to allow user to easily associate a skeleton with its scatterplot point. Alternatively, the mouse can be hovered over a scatterplot point which will cause the skeleton associated with it to be highlighted by a red circle.

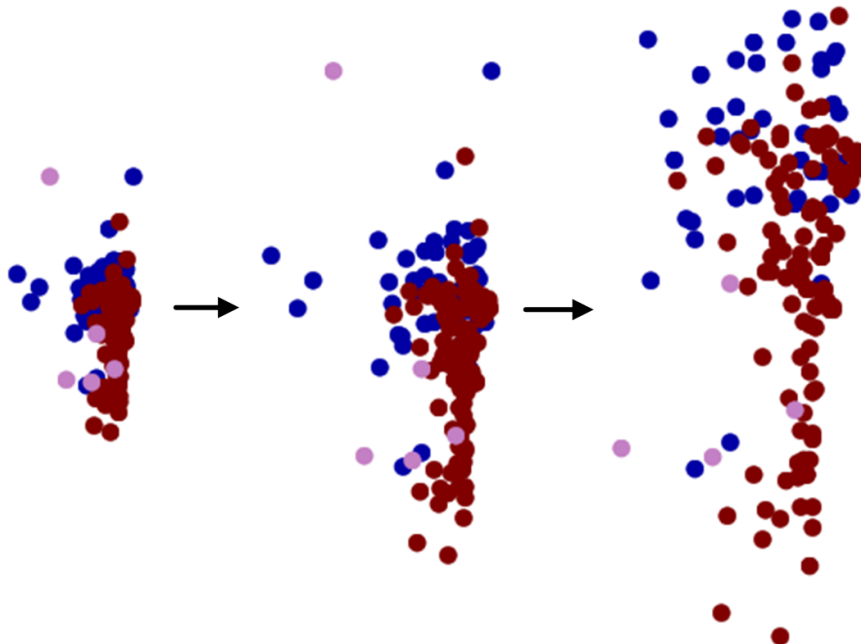


Fig. 4. The scatterplot view supports both translation and zooming in order to allow clusters of points to be investigated.

mocap sequences in a mocap database. MDS is then applied to the dissimilarity matrix in order to allow its structure to be visualized using a 2-dimensional scatterplot.

To facilitate understanding of this structure, motion classes and sub-classes are colour coded to allow similar motions to be easily identified. Choice boxes below the scatterplot view allow the user to select which sub-classes they are interested in viewing. The scatterplot view supports translation and zooming in order to allow clusters of points to be investigated as illustrated in Figure 4. A toolbar below the scatterplot view allows the user to switch between different motion normalization techniques (Section 4.2), MDS techniques (Section 5), and similarity metrics (Section 4.4). The toolbar also allows the user to return to the default view where all points are guaranteed to be visible and to toggle on or off tooltips that indicate the name of the mocap file associated with a scatterplot point.

### 3.2 Details-on-demand views

MotionVis consists of three details-on-demand views: the skeletal view, the motion map view, and the dissimilarity matrix view. A critical aspect of the proposed environment is the tight coupling between the scatterplot view and these details-on-demand views. Each of these views provides information useful for understanding the structure displayed in the scatterplot view.

#### *Skeletal view*

Selecting a point in the scatterplot view causes the mocap sequence associated with this point to appear in the skeletal view. The mocap sequence is animated using a simplified geometric model. Menu options allow the user to select either a stick-figure, cylinder, or ellipsoid-type model depending on their personal preference and available processing power. Below the skeletal view, standard movie controls are provided to allow the user to navigation through a mocap sequence. The user can navigate within the skeletal view using a world-in-hand navigation model or switch between a number of predefined camera views (*i.e.*, isometric, front, side, top) provided on the toolbar. The toolbar also provides the ability to toggle on or off the following features:

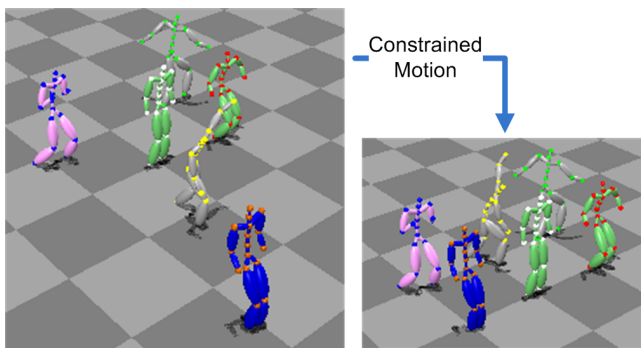


Fig. 5. A skeleton can be constrained to a single position in the virtual world in order to aid in comparing multiple mocap sequences.

- *Constrained motion*: it is natural to visualize a mocap sequence where the skeleton's position in the virtual world is unconstrained. However, when multiple mocap sequences are being considered, it is often easier to compare them when the skeletons are constrained to stay at a specific point within the virtual world (see Figure 5).
- *Shadows*: shadows provide a strong cue as to the position of a skeleton in the virtual world. Since realistic shadows are not required to establish position within a scene (see [9], page 279) we make use of a simplified shadow model in order to reduce the computational demands of our system. To further reduce computational demands, shadows can be turned off.
- *Path tracing*: the path followed by a skeleton is displayed as a curve (see Figure 6). This feature is particularly helpful when combined with the constrained motion feature as it allows the overall movement of a motion to be understood.
- *Shape encoding*: in order to associate a skeleton with its scatterplot point shape encoding is used as shown in Figure 3.

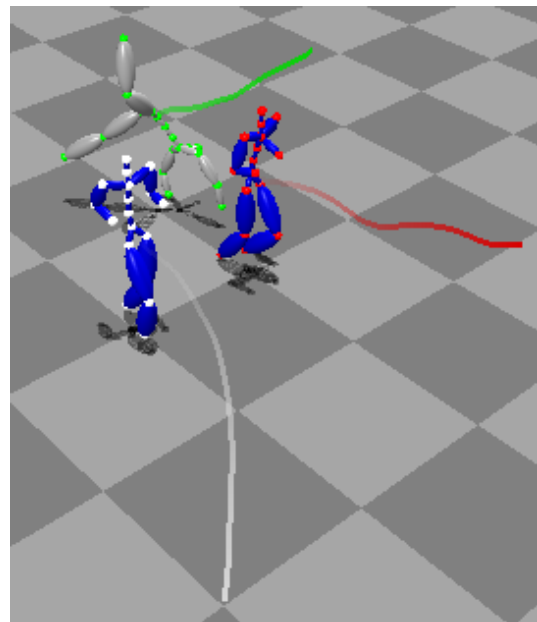


Fig. 6. Path tracing produces a curve that indicates the path traveled during a mocap sequence.

The association between a skeleton and scatterplot point can also be determined by hovering over either of these elements. Hovering over a scatterplot point causes a red circle to be drawn around the associated skeleton. Similarly, hovering over a skeleton causes a red circle to be drawn below it (indicating it has been selected) and the associated scatterplot point to be surrounded by a black square (see Figure 7).

#### *Motion map view*

The motion map view supports the same features and coordinated view techniques as the skeletal view. However, instead of laying out skeletons along the ground plane, they are laid out in a vertical matrix as shown in Figure 8. The major

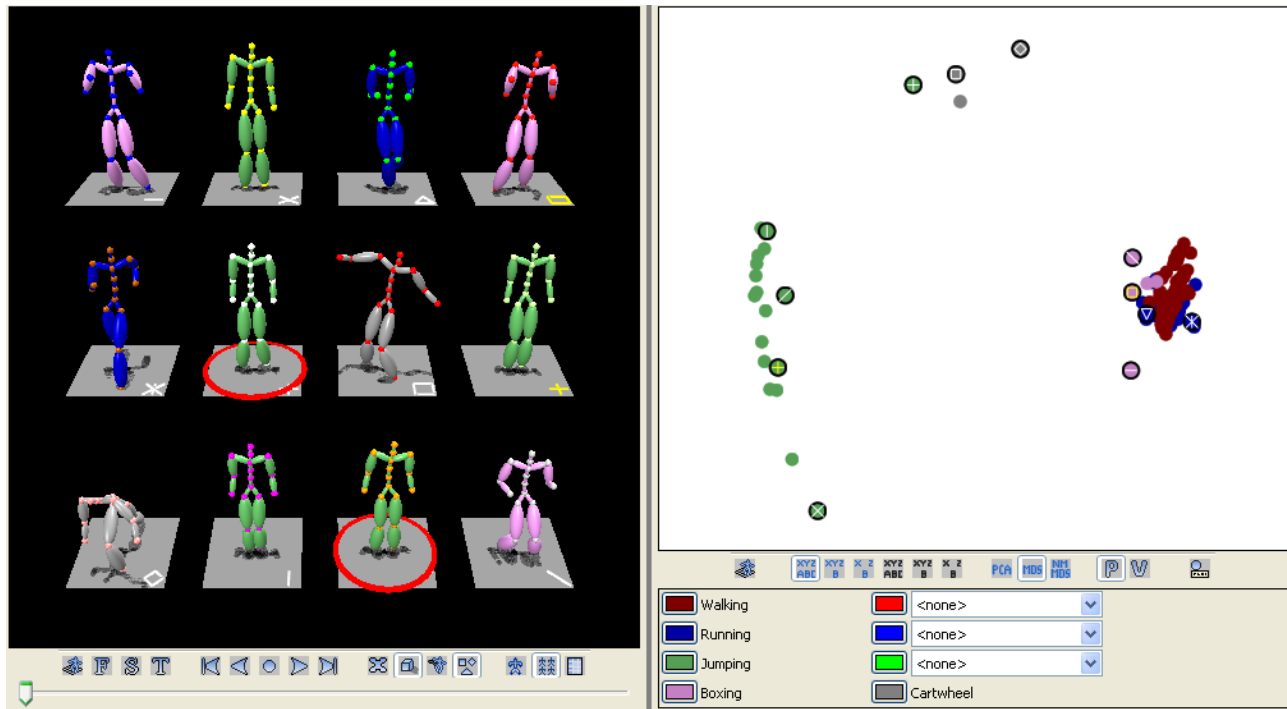


Fig. 8. The motion map view is designed to support the visualization of a large number of mocap sequences. Skeletons are laid out in a manner that ensures they never occlude each other.

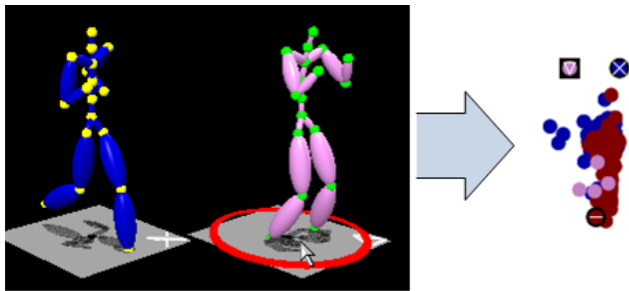


Fig. 7. Placing the mouse over a skeleton causes it to be selected (indicated by the red circle) and the point associated with it to be highlighted with a black square.

advantage of this layout is that it ensures skeletons will never occlude each other. However, this is at the expense of having to constrain all skeletons to a single point in space which can make understanding their motion more challenging. As discussed above, this limitation can be partially overcome by turning on path tracing.

#### *Dissimilarity matrix view*

The dissimilarity matrix view is divided into two sections (see Figure 9). In the top section, a dissimilarity matrix for the selected scatterplot points is given. Hovering over a point in the scatterplot view causes the row associated with this point to be highlighted. Selecting a cell in the dissimilarity matrix causes a line to be drawn between the two points corresponding to this cell and causes all points which are not selected to fade into the background. The colour of the

line indicates the distance between the points as specified by the colourmap in the bottom section of this view. The bottom section consists of a colourmap and a pair of spin controls which we collectively refer to as the *dissimilarity range control*. This control allows a range of dissimilarities to be specified. Pairs of points that have a dissimilarity within this range will have a line drawn between them (see Figure 17).

## 4 MOTION CAPTURE DATA

We begin this section with a discussion of the mocap data used to evaluate our system. Methods for normalizing this data are then presented. Finally, we discuss the skeletal model used in this work along with the mocap-based similarity metric considered in our evaluation.

### 4.1 Motion capture data

To evaluate the proposed system, we consider walking, running, jumping, boxing, and cartwheel mocap sequences from the CMU mocap database. These motion classes were selected because they form an interesting subset of the database. In particular, walking and running are similar in nature (arguably different degrees of a single locomotion class) whereas jumping, boxing, and cartwheel sequences are logically distinct motion types. There are also interesting subclasses within three of these motion classes. Specifically, we consider a 110 walking sequences (with subclasses “slow walk”, “walk”, and “walk” with various turns), 44 running sequences (with subclasses “run/jog”, “run”, and “run” with various turns), 17 jumping sequences (with subclasses “forward jumping”,

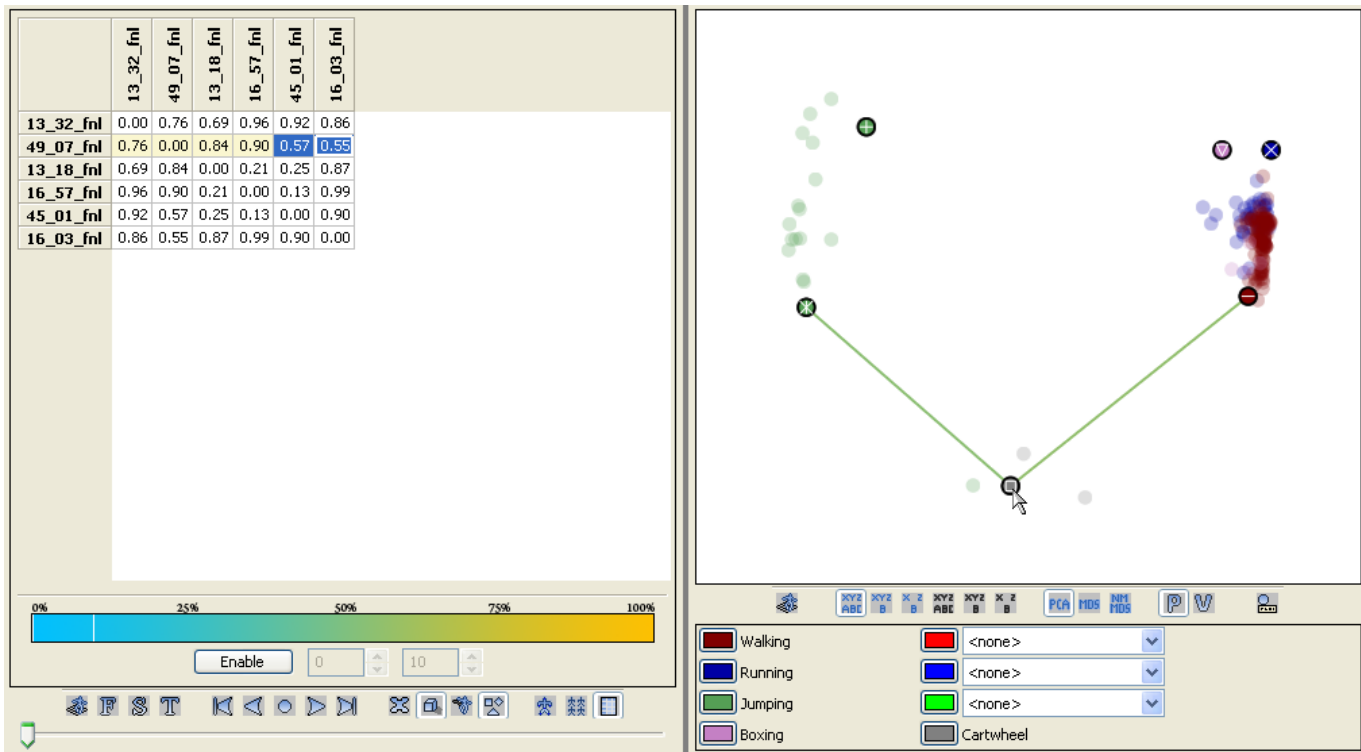


Fig. 9. The dissimilarity matrix view consists of two sections. In the top section, a dissimilarity matrix for the selected scatterplot points is given. The dissimilarity range control in the bottom section can be used to connect all pairs of points within a user specified range of dissimilarities by a line.

“jumping”, “high jump”, and “jump up and down, hop on one leg”), 5 boxing sequences, and 3 cartwheel sequences.

We now introduce some terminology that is useful for discussing mocap data. Human motion can be described using a simplified model of the human skeleton. A skeleton consists of *bones* that are connected by *joints*. Motion capture can be thought of as a process which records a temporal sequence of 3-dimensional joint positions. The position of all joints at a given time is known as a *pose*. A pose can be described as a vector  $p \in \mathbb{R}^{3 \times |J|}$ , where  $|J|$  is the number of joints in the skeletal model and each joint requires 3 elements to describe its 3-dimensional position. A *mocap sequence* can then formally be described as a time-dependent sequence of poses. This can be represented by a 2-dimensional matrix  $S \in \mathbb{R}^{T \times (3 \times |J|)}$ , where  $T$  is the number of poses (frames) in the mocap sequence.

The mocap sequences we are considering vary in length from 2 to 46 seconds (*i.e.*,  $T$  varies significantly between mocap sequences) and are sampled at a fixed rate of 120 frames per second. The CMU mocap data uses a skeletal model of 32 joints (*i.e.*,  $|J| = 32$ ). However, as will be discussed in Section 4.3, we make use of a simplified model that consists of only 18 joints.

## 4.2 Motion normalization

In general, we want two mocap sequences to be considered similar if the only significant difference between them is some global affine transformation, such as a translation or

rotation. As such, it is typical to normalize mocap data before measuring similarity. We illustrate the need for normalization by example. Consider two walking sequences which begin at different positions and orientations. To compare these motions, we should normalize the sequences so they start at the same position and have the same initial orientation. This is reasonable unless global position and orientation have special significance in the intended application.

For some applications, this single global normalization step is sufficient. However, we may desire a broader definition of similar. For example, we may desire all walking sequences to be considered similar regardless of whether they curve to the right or left. To achieve this, we can normalize the sequences every frame so the walking motion is constrained to be along a line facing in a particular direction.

Whether normalization is performed only once or at each frame, it is typically applied uniformly to all joints. We denote per frame normalization with an  $F$ , a single initial normalization with an  $I$ , and the coordinates  $(x, y, z)$  and angles  $(\alpha, \beta, \gamma)$  to be normalized with subscripts. Figure 10 gives the coordinate system used by MotionVis. MotionVis supports the following normalization methods:

- $F_{xyz, \alpha\beta\gamma}$ : at each frame the position  $P_{xyz}$  and orientation  $\theta_{\alpha\beta\gamma}$  are normalized. This causes the similarity of motions to be based purely on the location of joints relative to the root joint (see Figure 11). Informally, this causes all motions to be constrained to a single position and a fixed orientation which removes any differences due to motions following different paths.

- $F_{xy,z,\beta}$ : rotations about the x and z-axis are no longer normalized. This has little effect on most motion classes since there is generally little rotation about these axes. However, it is significant for a motion class such as cartwheel where there is significant rotation about these axes.
- $F_{xz,\beta}$ : movement along the y-axis is no longer normalized. This has little effect on most motion classes since there is generally little movement along this axis. However, it is significant for a motion class such as jumping which has significant movement along the y-axis.
- $I_{xyz,\alpha\beta\gamma}$ : mocap sequences are normalized to begin at the same position and to have the same initial orientation.
- $I_{xy,z,\beta}$ : mocap sequences are normalized to begin at the same position and to have the same orientation about the y-axis.

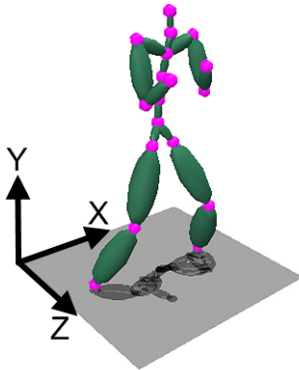


Fig. 10. MotionVis uses a right-handed coordinate system where the the ground plane is perpendicular to the y-axis.

In order to compare mocap sequence, it is typically desirable to map the mocap sequences to a common skeletal model. We adopt this approach here.

### 4.3 Skeletal model

All mocap sequences in the CMU mocap database use a skeletal model consisting of 32 joints. The placement of markers used to record these mocap sequences can be found on the CMU mocap website<sup>3</sup>. This model is relatively detailed. For example, the skeletal model considers the position of the thumb joint. This level of detail is unnecessary when comparing motion classes and may even be misleading unless the significance of different joints is explicitly considered by the similarity metric. For example, when comparing walking mocap sequences, changes in the position of the thumb joint are clearly less important than changes in position of the femur joint. To evaluate our system, we consider the similarity metric proposed by Li and Prabhakaran [7] which does not perform any joint weighting. For these reasons, we make use of a simplified model consisting of the 18 most significant joints as shown in Figure 11. This should result in a skeletal model

similar to that considered by Li and Prabhakaran who also make use of a skeletal model with 18 joints, although do not specify which joints were used.

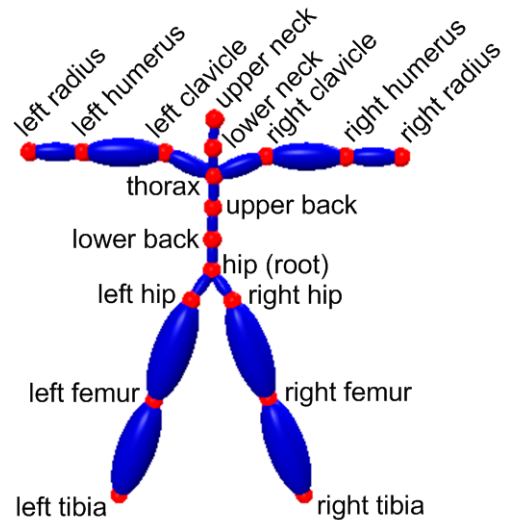


Fig. 11. Skeletal model consisting of rigid *bones* that are connected by *joints*. This model considers the 18 most significant joints in the CMU mocap data.

### 4.4 Li's similarity metric

In this section we summarize the similarity metric proposed by Li and Prabhakaran [7] using the notation introduced in Section 4.1. Li's similarity metric consists of the follow 3 steps:

- 1) Apply singular value decomposition (SVD) to each mocap sequence  $S_i$  to find its principal component,  $c_i$ .
- 2) Normalize each principal component,  $c_i$ , so its first element is non-negative by multiplying  $c_i$  by  $-1$  if and only if the first element is negative.
- 3) The dissimilarity of two mocap sequences,  $S_i$  and  $S_j$ , are given by  $diss(S_i, S_j) = |c_i \cdot c_j| = |c_i| |c_j| |\cos(\theta)|$ , where  $\theta$  is the angle between the two principal components.

Figure 12 illustrates how Li's similarity metric works. Consider two mocap sequences,  $S_1$  and  $S_2$ , of dimensionality 2 (instead of the 54 dimensions required for a skeleton of 18 joints), where the first motion consists of 4 poses and the second motion consists of 3 poses. Each pose can be viewed as a data point in a 2-dimensional space. The principal component of a mocap sequence is the axis which captures the greatest variance of the data when the data is projected onto this axis (the blue and red lines in Figure 12). The premise of Li's similarity metric is that similar mocap sequences should have similar principal components. As such, the dissimilarity between two mocap sequences is defined as the smallest angle between the principal components of these mocap sequences. The normalization in step 2 is required because the SVD of a matrix is only uniquely determined up to the sign.

This similarity metric has many favourable properties which has encouraged its use in this work:

3. mocap.cs.cmu.edu/markerPlacementGuide.pdf



- it can be applied to mocap sequences with varying numbers of poses,  $T$
- it is relatively inexpensive to compute
- it is simple to understand and implement

We also consider a simple alternative based on Li’s similarity metric. Instead of applying the metric to joint positions, we apply it to joint velocities. The velocity of each joint is estimated as the difference in position between two consecutive poses. Li’s similarity metric can be applied to the resulting velocity matrix without modification.

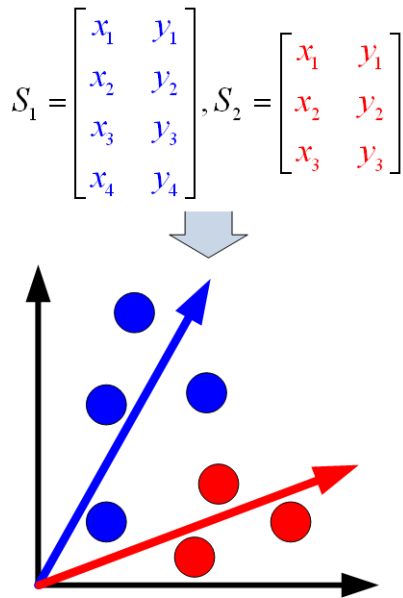


Fig. 12. Li’s similarity metric applied to artificial mocap sequences consisting of 4 poses (blue) and 3 poses (red), where each pose has a dimensionality of 2. The corresponding colour axes indicate the principal component for each of these matrices and the angle between these axes is used as a measure of dissimilarity.

## 5 DIMENSIONALITY REDUCTION

A mocap-based similarity metric can be used to construct a dissimilarity matrix which indicates the dissimilarity between all pairs of mocap sequences in a mocap database. We wish to construct a visualization of this dissimilarity matrix, so we can investigate the structure imposed on the database by the similarity metric. Understanding this structure will allow key aspects of the similarity metric to be understood. Our approach is to apply MDS to the dissimilarity matrix so it can be visualized as a 2-dimensional scatterplot.

Our MotionVis environment supports three MDS techniques that have different strengths and weaknesses:

- 1) *Classic MDS* [19]: a linear dimensionality reduction technique which is relatively fast to compute. This approach also supports forward and reverse mappings. Depending on the final application, a linear subspace may be preferred and thus understanding the linear structure imposed on a database by a similarity metric will be of particular interest.

- 2) *Metric MDS* [19]: a nonlinear dimensionality reduction technique. It is more computationally expensive than classic MDS and does not provide mapping functions. However, the resulting visualization is generally more accurate since it performs a nonlinear reduction.
- 3) *Non-metric MDS* [23]: a technique that aims to preserve the rank order of dissimilarities as opposed to the exact dissimilarity. The resulting visualization will reflect the rank order of dissimilarities better than the above methods, at the expense of losing specific information about the dissimilarity between pairs of mocap sequences. Its computational cost is comparable to metric MDS and it also lacks mapping functions. This visualization is ideal when only rank order is of interest. In practice, we see this visualization being used in conjunction with the metric MDS visualization or as a “last resort” when the stress, as defined below, of metric MDS is too large for its visualization to be trusted.

It is critical that the scatterplot be an accurate visual representation of the dissimilarity matrix. We make use of Kruskal’s stress metric [23] in order to assess how well a visualization reflects the dissimilarities in the dissimilarity matrix. This metric is a simple normalization of the sum of squared differences between the Euclidean distances in the visualization,  $x_{ij}$ , and the actual input dissimilarities,  $d_{ij}$ :

$$stress = \sqrt{\frac{\sum \sum (x_{ij} - d_{ij})^2}{\sum \sum d_{ij}^2}}$$

High stress indicates that the resulting visualization poorly reflects the dissimilarity matrix. Table 1 gives the stress for each MDS technique for the different motion normalization methods discussed in Section 4.2. Roughly speaking, a stress of 0.078 means that on average there is a 7.8% error between the actual dissimilarity and the Euclidean distance as indicated in the scatterplot visualization. We suggest only using visualizations where the stress is less than 0.1 and as such provide this stress measure directly to the user.

	$F_{xyz,\alpha\beta\gamma}$	$F_{xyz,\beta}$	$F_{xz,\beta}$	$I_{xyz,\alpha\beta\gamma}$	$I_{xyz,\beta}$
Classic MDS	0.078	0.143	0.106	0.218	0.213
Metric MDS	0.052	0.092	0.068	0.143	0.156
Non-metric MDS	0.035	0.086	0.064	0.053	0.070

TABLE 1

Stresses of different MDS techniques for different motion normalization methods.

## 6 DESIGN OF MOTIONVIS

This section justifies the design discussions made during the development of MotionVis. Specifically, we discuss our choice of visual encodings, layout algorithms, navigation models, use and selection of details-on-demand views, and use and implementation of coordinated visualization.

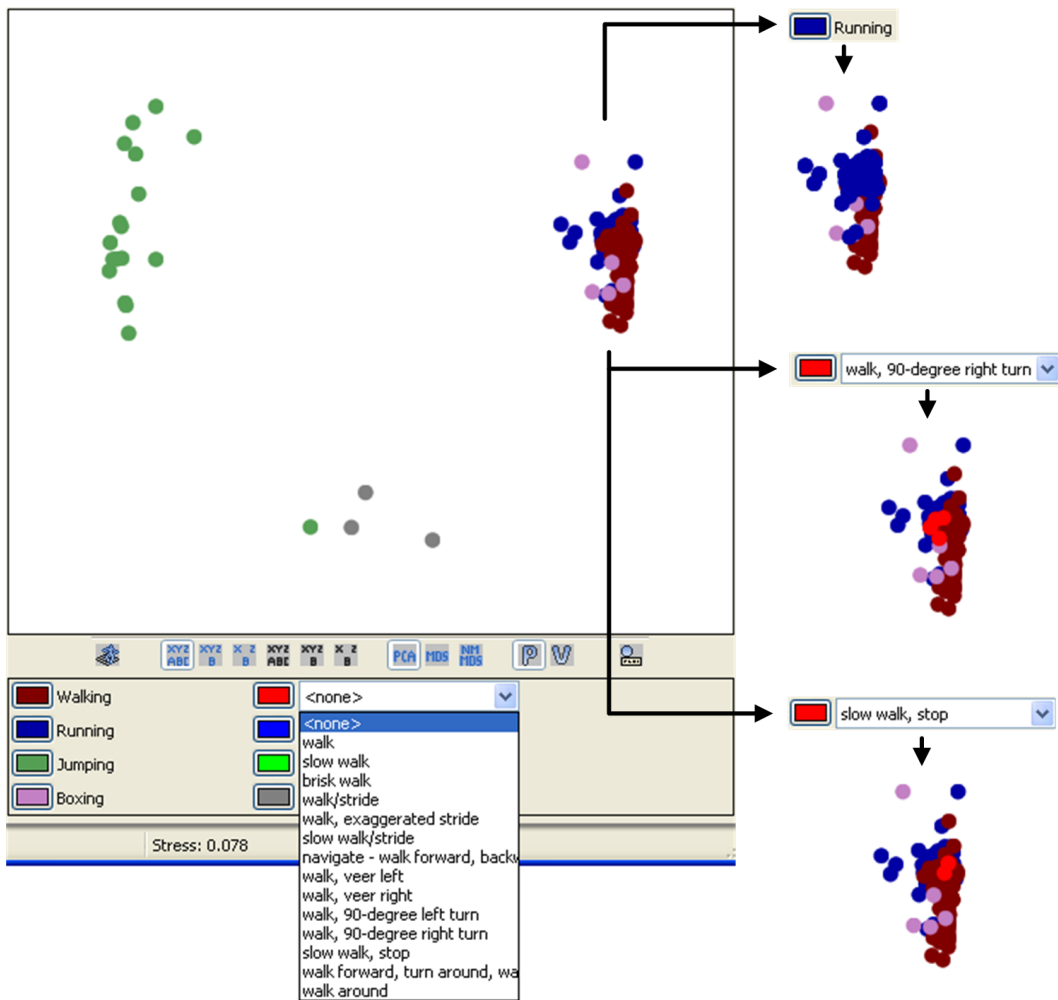


Fig. 13. The scatterplot view uses colour to quickly allow related motion (sub)classes to be identified. Choice boxes allow specific subclasses to be viewed. Clicking on the button next to a motion (sub)class ensures all points of this (sub)class within the current view are visible.

## 6.1 Scatterplot view

A critical design decision in our system is the use of MDS to allow the dissimilarity matrix to be visualized as a scatterplot. Since we are interested in understanding the similarity metric which defines the dissimilarity between pairs of data points, we selected a visualization that focuses on the relationship between data points. This rules out the use of high-dimensional visualization techniques such as scatterplot matrices, star glyphs, and parallel coordinates which are intended to reveal the relationship between dimensions, but fail to give a clear indication of the relationship between pairs of data points.

The relationship between data points is often referred to as proximity data. Two common methods exist for visualizing proximity data: graph layout algorithms and dimensionality reduction. We have selected dimensionality reduction and specifically MDS as it directly aims to preserve the dissimilarity between pairs of data points [19]. In addition, MDS is a family of dimensionality reduction techniques [24] that allow a range of useful visualizations to be obtained as discussed in Section 5. DeJordy [18] argues that graph layout algorithms often give superior visualizations of proximity data as they

are less sensitive to outliers and are better suited to interactive exploration. Unfortunately, these advantages are offset by the fact that a graph layout based visualization will not be reflective of the actual dissimilarity between data points. As such, we prefer the use of MDS, but are considering the inclusion of a graph layout based visualization as a future extension to MotionVis.

### Visual encoding

In general, the most important aspect of a similarity metric is its ability to distinguish between different motion classes. The scatterplot view indicates the global structure imposed on a mocap database by a similarity metric. As such, the most critical encoding in our system is distinguishing between different motion classes in the scatterplot view.

We encode the motion classes using hue since Mackinlay [21] has established it as being the best encoding for nominal data (next to spatial position). Berlin and Kay [22] conducted a study of more than 100 languages and determined that there is a relatively fixed order in which colour names appear in languages. Ware [9] has suggest a set of colours to be used for

visual encoding based on the 11 most common colours found in the Berlin and Kay study. The order of colours suggested by Ware is largely based on colour space research. We have adopted Ware’s colour set with a few small modifications to ensure our application is suitable for colour blind individuals (see Appendix A).

### Guaranteed visibility and sub-class highlighting

Points in the scatterplot view are often tightly cluster. A method to ensure that all points of a given motion class are visible is required. This is achieved by having the user click on a button next to the motion class to ensure all points of that class are visible.

To facilitate understanding the structure within a motion class, different subclasses can be selected from choice boxes below the scatterplot view. Selecting a subclass causes all points from this subclass to be drawn in a more saturated colour than the class it belongs to and also ensures these points will be rendered last.

## 6.2 Skeletal view

In order to fully understand the structure in the scatterplot view, it is typically necessary to investigate the mocap sequences associated with various scatterplot points. An effective means to visualize a mocap sequences is direct visualization using a (simplified) animated human model since we are extremely adept at understanding human motion. We refer readers to the work of Johansson [11] who demonstrated that humans can interpret human motion even from simple point-light displays. Equally as convincing is the widespread use of commercial and open source animation packages which make use of simple geometric human models during the development of realistic, human animation sequences. Based on the success and widespread use of commercial and open source animation packages, we have decided to adopted many of the conventions used by these programs (*e.g.*, world-in-hand navigation combined with specific preset views, use of a tiled ground plane to indicate depth).

Hodgins *et al.* [12] conducted experiments with a simple stick figure model and a more detailed polygonal model in order to establish that a viewer’s perception of motion characteristics is affected by the geometric model used. Their results suggest that a subject is better able to detect small motion changes with the polygonal model than the stick figure model. However, they caution against generalizing these results beyond the three types of motion variations they considered. Nevertheless, these results do establish that the perception of motion is influenced by the geometric model used to render the motion. For this reason, we allow the user to select between a simple stick figure model and more complex cylinder and ellipsoid-based models. As a practical matter, these three model types also provide a compromise between frame rate and model complexity.

### Visual encoding

To help establish the motion class associated with a skeleton, we decided to have the colour of a skeleton’s bones reflect

the motion class it belongs to (see Figure 3). This has the additional advantage that the visual encoding of the motion classes uses colours with well established names as discussed in Section 6.1. This is a great aid when trying to discuss different mocap sequences in the skeletal view. To distinguish between different skeletons from the same motion class, we ensure that the colour of a skeleton’s joints are unique. The first ten joint colours are assigned based on the colours suggested by Ware [9]. This colour list is also used to encode the bones of a skeleton. Although it is not critical, we have modified the saturation value used for joint colours to make them different than the bone colours. In practice, a user rarely has a need to consider more than ten skeletons, but if more skeletons are considered their joint colour are assigned a random colour of high saturation (to ensure it does not blend in with the black background) that is guaranteed to be unique.

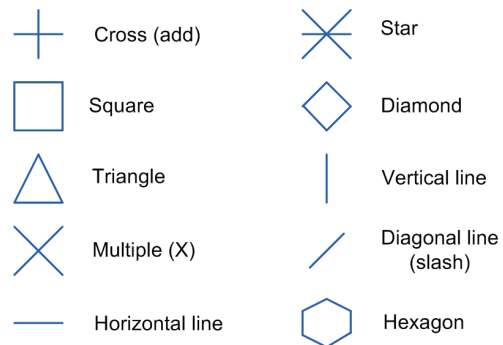


Fig. 14. Shapes used to associate selected scatterplot points with skeletons.

In order to allow the user to associate a scatterplot point with its skeleton, a shape encoding scheme is employed (see Figures 3 and 8). We use shapes for this encoding because Mackinlay [21] has demonstrated that shapes are a strong encoding technique for nominal data. Specifically, we use the set of shapes shown in Figure 14 as they are easily distinguishable from each other and have well known names. If more than ten mocap sequences are considered, we repeat the shape list, but change the colour.

### Coordinated visualization

The value of coordinated views for analyzing data is well studied in InfoVis (see [25] for a recent review). We make use of coordinate visualization in order to allow user to quickly determine the association between a skeleton and scatterplot point by hovering over either of these elements. Hovering over a scatterplot point causes a red circle to be drawn on the ground plane around the associated skeleton. A red circle is used to indicate the associated skeleton since red has good contrast with both the floor and background and the circle encompasses the skeleton without obstructing one from visualizing its motion. When the user hovers over a skeleton a red circle is drawn below it and the associated scatterplot point is surrounded by a black square (see Figure 7). Selecting a skeleton can be challenging when it is in motion and thus we have decided to indicate a *successful* selection by drawing the red circle. A black square is used to indicate the

associated scatterplot point as the square will be preattentively processed since it is the only square within a collection of circles ([9], Chapter 5). In practice, it is often useful to use hovering even if shape encoding is turned on, since it is faster to hover the mouse over a point or skeleton than to search for corresponding shapes.

### Skeletal view layout

An important design decision is how to layout skeletons within the skeletal view. We have opted for the circular pattern illustrated in Figure 15, as it maximizes the number of skeletons that can be visualized within the skeletal view. An alternative that we considered was to layout skeletons along lines of a fixed length. Once a line was filled, a new line would be started either behind or in front of the previously filled lines. Although this layout is simpler in many respects, it is difficult to select a suitable line length. If the line length is too small, too many lines will be required when a large number of skeletons is selected. If the line length is too large, it becomes difficult to see all the skeletons at once. The circular layout pattern avoids the need to decide on a suitable line length and is optimal in the sense that it allows as many skeletons as possible to be visualized at a given zoom level.

The spacing between skeletons is set large enough that they will not interfere with each other when skeletons are constrained to their initial position. The spacing of our tiled floor reflects this spacing so that skeletons are laid out at the corners of the tiles. This is visually pleasing and aids the user in establishing where a skeleton will be placed next.

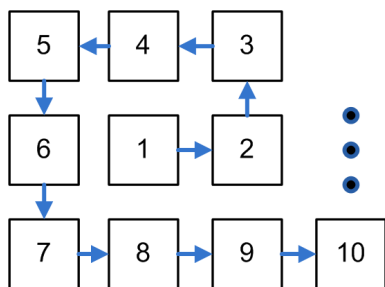


Fig. 15. Skeletons are laid out in the counter-clockwise spiral pattern indicated by the arrows. This layout pattern can support a large number of skeletons within a small area.

### 6.3 Motion map view

It is often inconvenient to visualize more than a few mocap sequences in the skeletal view since skeletons begin to occlude each other. We have developed the motion map view in order to support the visualization of large numbers of mocap sequences. This view uses the same visual encodings, coordination techniques, and layout algorithm as the skeletal view, except skeletons are laid out in the xy plane instead of the xz plane in order to ensure they never occlude each other (see Figure 8). The major limitation of this motion map view is that all skeletons are constrained to a single point in space which can make visualizing their motion less natural than in

the skeletal view. As discussed in Section 3.2, this limitation can be partially overcome by turning on path tracing.

### Navigation model

Navigation in the motion map view uses a multi-view, world-in-hand navigation model. Each skeleton is governed by a world-in-hand model where the centre point of the world corresponds to the root node of the skeleton. These world-in-hand models are linked in the sense that all rotations and translations are applied uniformly to the skeletons. This facilitates the comparison of motions since all skeletons are viewed from the same viewpoint. Like the skeletal view, a number of common predefined views are provided (see Figure 16).

### 6.4 Dissimilarity matrix view

The dissimilarity matrix view complements the skeletal and motion map views by providing a quantitative method for exploring the relationship between scatterplot points on both a local and global scale. A dissimilarity matrix for the selected scatterplot points is given in the top section of the view. This dissimilarity matrix can be used to quickly gain a quantitative understanding of the relationship between the selected points.

The dissimilarity range control in the bottom section of this visualization allows a range of dissimilarities to be selected. A line will then be drawn between all pairs of points that have a dissimilarity within the selected range, as shown in Figure 17. The colour of a line indicates the dissimilarity between a pair of points as specified by the colourmap. This control allows global information about the dissimilarity between clusters to be quickly determined.

### Coordinated visualization

Tight coupling between the dissimilarity matrix view and the scatterplot view allows a user to rapidly gain a more quantitative understanding of the relationship between pairs of points. Hovering over a point in the scatterplot view causes the row associated with this point to be highlighted. Selecting a cell in the dissimilarity matrix causes a line to be immediately drawn between the two points corresponding to this cell and causes all points which are not selected to fade into the background (see Figure 9).

### Colourmap selection

Care should be taken when creating a colourmap to ensure it will produce perceptually meaningful results. Here we make use of a colourmap consisting of desaturated colour for two reasons. First, the number of lines drawn for a given user selected dissimilarity range is often large which causes large portions of the scatterplot view to be filled by lines. Tufte [26] recommends the use of desaturated colours for large regions. More specific to our environment, is that these colours are distinct from those used to encode the motion classes. This makes it easy to distinguish scatterplot points from lines.

Currently, our colourmap has not been designed to be perceptually uniform. This would be a desirable improvement we plan to incorporate in a future version. We are also considering the use of a segmented colourmap.

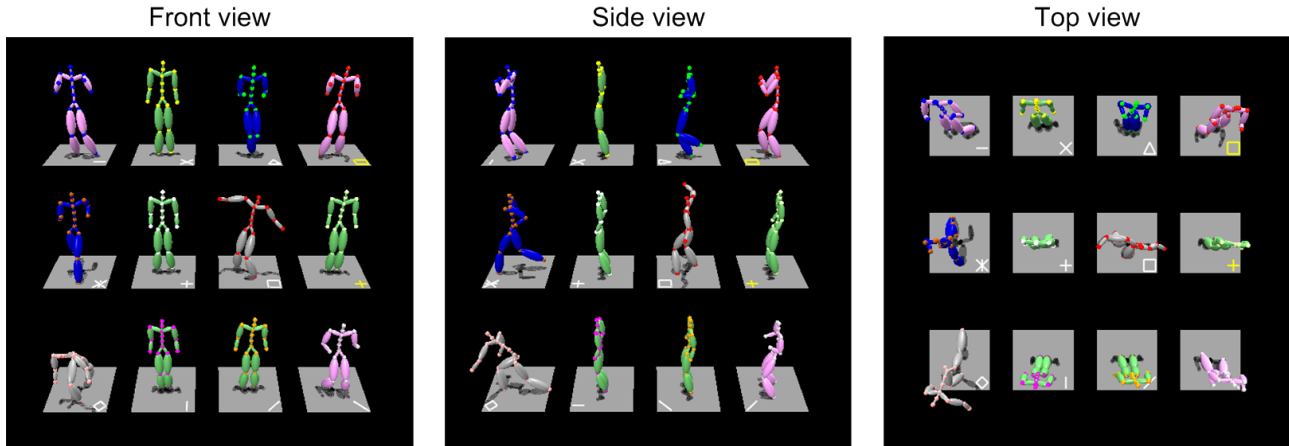


Fig. 16. Predefined views (excluding isometric view) supported by the motion map view. A multi-view, world-in-hand navigation model is used to facilitate the easy comparison of motions. Notice that in the top view, the top of all skeletons are shown as opposed to just the top row of the 4x3 grid as would be the case in a standard world-in-hand navigation model.

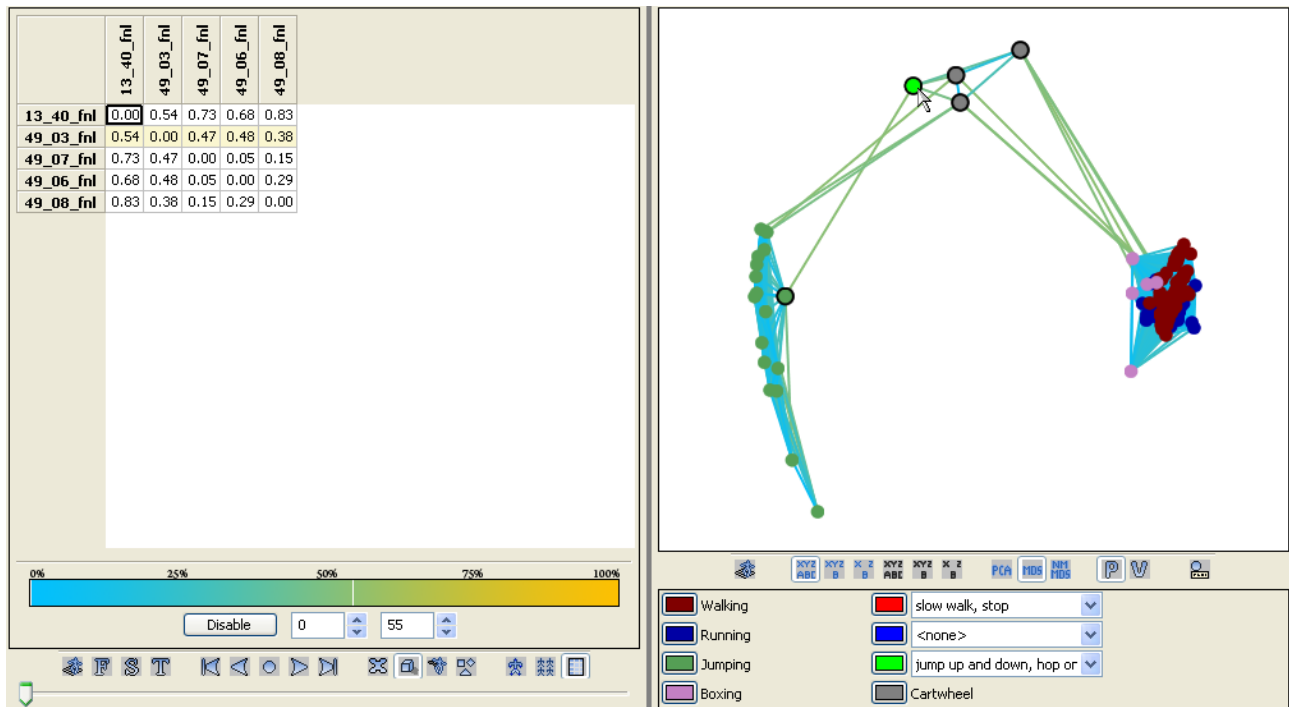


Fig. 17. The dissimilarity matrix view can be used to gain a better understand of both the global and local structure of the scatterplot view. A dissimilarity matrix corresponding to the selected scatterplot points is given in the top section of the dissimilarity matrix view, which indicates the dissimilarity between all pairs of selected points. This view can also be used to gain an understanding of the global relationship between pairs of points. This is accomplished by using the dissimilarity range control to select a range of dissimilarities. In this example, the user is visualizing all pairs of points that have a dissimilarity between 0 and 55 and has successfully established which jumping mocap sequence is closest to the outlier jumping sequence highlighted in bright green.

## 6.5 Implementation details

MotionVis has been implemented in C++ and makes extensive use of OpenGL<sup>4</sup> and wxWidgets<sup>5</sup> for implementing the graphical user interface. We have taken advantage of two open source mocap players, MotView<sup>6</sup> and wxMotionViewer<sup>7</sup>, as an initial codebase that supports loading and animating a mocap file. This codebase has been extensively modified to suit our requirements (*e.g.*, to support the loading and animation of multiple skeletons, to provide more computationally efficient rendering of animations, to allow bone and joint colours to be specified), but we are indebted to the authors of these programs for their excellent work. MDS is performed using the *mdscale* command provided in MATLAB<sup>8</sup> which supports classic, metric, and non-metric MDS.

The scatterplot, motion map, and dissimilarity metric views have all been implemented by the author using OpenGL and wxWidgets, whereas the skeletal view borrows heavily from MotView and wxMotionViewer. The coordination between the scatterplot and the details-on-demand views is solely the work of the author. The author also implemented Li's similarity metric along with the code required to perform motion normalization.

The Mocap files used for evaluating our system come from the CMU mocap database. This database supports a number of different file formats. Unfortunately, files are not provided in Biovision's BVH format, which is used by most commercial and open source mocap players, including MotView and wxMotionViewer. As such, we have made use of the freely available Amc2Bvh program<sup>9</sup>, in order to convert Mocap files in the CMU mocap database to BVH format. Unfortunately, this program skips the lower-back joint when converting the file. We have implemented code that corrects for this omission and have notified the author of Amc2Bvh of this bug.

## 7 SCENARIOS OF USE

A number of scenarios of use are explored in this section. We begin by illustrating the use of MotionVis for analyzing the overall structure imposed on a mocap database. Use of MotionVis for understanding outliers is then discussed. We then consider the effects of different motion normalization methods on Li's similarity metric before discussing our proposed extension, which considers joint velocities.

### 7.1 Analyzing the overall structure

The overall structure imposed on a mocap database by a similarity metric can be largely understood using just the scatterplot view along with the features that operate on this view. Upon loading MotionVis, the scatterplot view is set to its default viewpoint which guarantees all points are visible.

4. [www.OpenGL.org](http://www.OpenGL.org)

5. [www.wxWidgets.org](http://www.wxWidgets.org)

6. [www.cs.wisc.edu/graphics/Courses/cs-838-2000/Students/gardner/motView/](http://www.cs.wisc.edu/graphics/Courses/cs-838-2000/Students/gardner/motView/)

7. [cgg.ms.mff.cuni.cz/~semancik/research/wxmv/index.html](http://cgg.ms.mff.cuni.cz/~semancik/research/wxmv/index.html)

8. [www.mathworks.com](http://www.mathworks.com)

9. [vipbase.net/amc2bvh/](http://vipbase.net/amc2bvh/)

Figure 13 is used to illustrate investigating the overall structure with MotionVis.

A user should begin by checking the *stress* of the visualization. In this example, the stress is 0.078 as indicated on the status bar. This is small enough that the visualization will be an accurate representation of the dissimilarity matrix.

From the default scatterplot view it is immediately noticeable that the similarity metric does a good job of clustering jumping (green) and cartwheel (gray) mocap sequences. However, it clearly cannot distinguish between walking, running, and boxing sequences. We can further investigate the global structure by clicking on the *running* button which ensures all points corresponding to running mocap sequences will be made visible as illustrated in Figure 13. This indicates that although the similarity metric has trouble distinguishing between running and walking sequences, the running sequences do form a tight cluster that overlaps only the top of the walking cluster.

It could be that the walking sequences near the top of the walking cluster correspond to walking motions that are in some sense similar to running (for example, people walking briskly). This can be investigated by exploring different walking subclasses as shown in Figure 13. Unfortunately, this investigation indicates that in fact the mocap sequences which overlap with the running sequences are from a diverse set of subclasses such as "walking, 90-degree right turn" and "slow walk, stop".

A good practice is to consider the scatterplot visualization produced by the other MDS embedding techniques to ensure these same conclusions hold (see Figure 17 for the scatterplot produced by metric MDS). When the stress is low (as is the case here), it is extremely unlikely that different conclusions will be drawn from these alternative visualizations. However, when the stress is high, more care must be taken.

In this example, examining the other scatterplot visualizations will simply confirm the user's previous conclusions. Depending on the application, this similarity metric may be reasonable and the user will begin to study the local structure resulting from the metric. If this global structure is unacceptable, the user may either select a new metric or start to investigate the local structure in order to try and gain a deeper insight into the current metric with the hopes of being able to improve it.

### 7.2 Analyzing outliers

Investigating outliers often given valuable information about a similarity metric. In this example, we will consider the sole jumping sequence that is located near the cluster of cartwheel mocap sequences as shown in Figure 17. It is useful to begin investigating an outlier by visualizing its mocap sequence in the skeletal view. This allows the user to become familiar with the exact nature of the mocap sequence. In this example, the mocap sequence is notable because it consists of a person that not only jumps up and down, but also hops on one foot. By selecting the "jump up and down, hop on one foot" subclass the user can quickly determine that this is the only jumping sequence of this nature.

At this point, a deeper investigation of the outlier can be performed using the dissimilarity matrix view. By selecting points near the outlier, local structure information can be quickly obtained using the dissimilarity matrix. Hovering the mouse over a point causes the row corresponding to this point to be highlighted. In the example given in Figure 17, it can easily be determined that the dissimilarity between the outlier and the cartwheel mocap sequences is between 0.38 and 0.48. Furthermore, it can be determined that the range of dissimilarity between the cartwheel motions is 0.05 to 0.29. The dissimilarity range control can then be used to establish that the jumping sequence closest to this outlier has a dissimilarity of 0.54.

The user has discovered that the reason for this jumping sequence being an outlier is related to it being the sole jumping sequence where the person hops on one foot. Although it is true that under this similarity metric this jumping sequence is more similar to the cartwheel motions sequences than any jumping sequence, the user has also established that it is not really a member of either cluster. The closest cartwheel sequence has a dissimilarity of 0.38 whereas the largest dissimilarities between any two cartwheel motions is only 0.29. The unique nature of this jumping sequence has caused it to be relatively dissimilar to all other mocap sequences.

### 7.3 Analyzing motion normalization methods

It is important to realize that the investigation performed in Section 7.2 was performed under a specific similarity metric and motion normalization method. Let us assume the goal of the user is to be able to distinguish the jumping and cartwheel sequences from the other mocap sequences. In this case, it may be reasonable to use the  $F_{xz,\beta}$  normalization methods as opposed to the  $F_{xyz,\alpha\beta\gamma}$  method considered above. Relaxing the constraint on the y-axis should cause jumping sequences to form a strong cluster since they are the only motion class that exhibits significant movement along this axis.

As indicated in Figure 18, using the  $F_{xz,\beta}$  normalization method does cause the jumping mocap sequences to form a tight cluster (including the outlier investigated in Section 7.2). The cartwheel sequences also form a unique cluster. Surprisingly, there is a single running mocap sequence near the jumping cluster. Figure 19 shows this running sequence in the motion map view. Notice that it is floating above the ground. This explains why it is near the jumping cluster, but more importantly indicates it is an erroneous motion file that needs to be corrected.

### 7.4 Analyzing the joint velocity similarity metric

The above scenarios of use have all used Li’s similarity metric. In this section, we consider a simple alternative to Li’s metric that uses joint velocity instead of joint position (see Section 4.4 for details). A significant limitation of Li’s similarity metric is that it does not distinguish between walking and running mocap sequences. A significant source of variation between these mocap sequences is the rate at which joints move. For this reason, we hypothesize that our velocity-based metric will allow these motion classes to be distinguished.

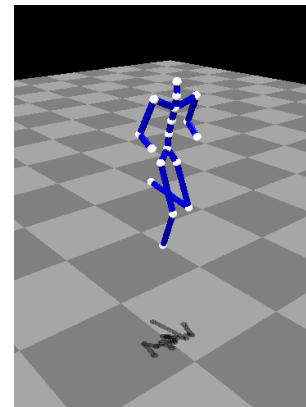


Fig. 19. An erroneous running sequence discovered using MotionVis.

Figure 20 shows the resulting scatterplot views for our velocity-based metric under a number of different motion constraints. As can be seen, it is *not* able to distinguish between walking and running motions. MotionVis allows this failure to be determined within seconds and, if desired, would allow the user to start investigating why the expected results were not obtained.

## 8 DISCUSSION

In this section, we begin by discussing the strengths of the proposed environment along with its limitations and some potential future extensions. We then explore the possibility of extending MotionVis into a general-purpose visualization environment for analyzing similarity metrics.

### 8.1 Strengths

The goal of this project is to design and develop an interactive environment to aid in the rapid understanding of the structure imposed on a mocap database by a given similarity metric. This goal has largely been achieved by taking advantage of established InfoVis practices:

- *Visual encoding*: we make use of visual encoding to allow motion (sub)classes to be quickly identified in both the scatterplot, skeletal, and motion map views. This is particularly powerful in the scatterplot view where the global structure imposed on a mocap database by a similarity metric can be quickly understood as illustrated in Section 7.1. In addition, shape encoding allows a user to quickly establish the association between selected scatterplot points and mocap sequences in the details-on-demand views.
- *Details-on-demand*: the skeletal and motion map views allow for the direct visualization and comparison of mocap sequences. The dissimilarity matrix view allows the dissimilarity between pairs of mocap sequences to be quickly understood on both local and global levels.
- *Coordinated views*: tight coupling between the scatterplot and details-on-demand views allows the user to quickly gain a deep understanding of the local structure in the

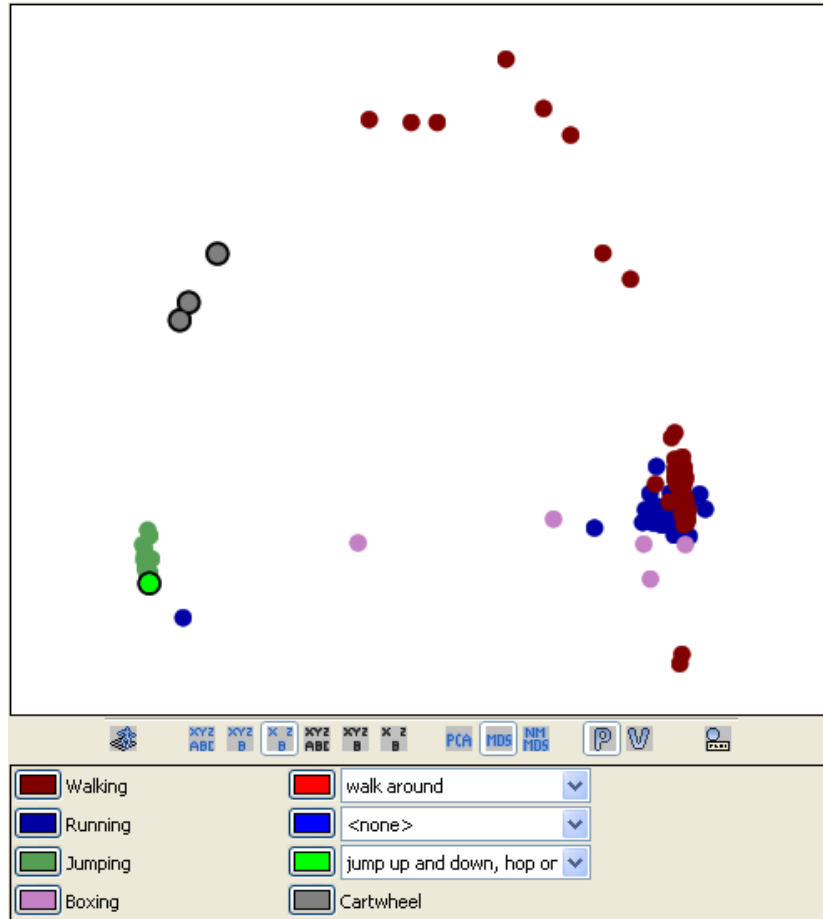


Fig. 18. Structure of the database under the  $F_{xz,\beta}$  normalization method. Notice how tightly clustered the jumping mocap sequences are since they are the only motion class with significant movement along the y-axis.

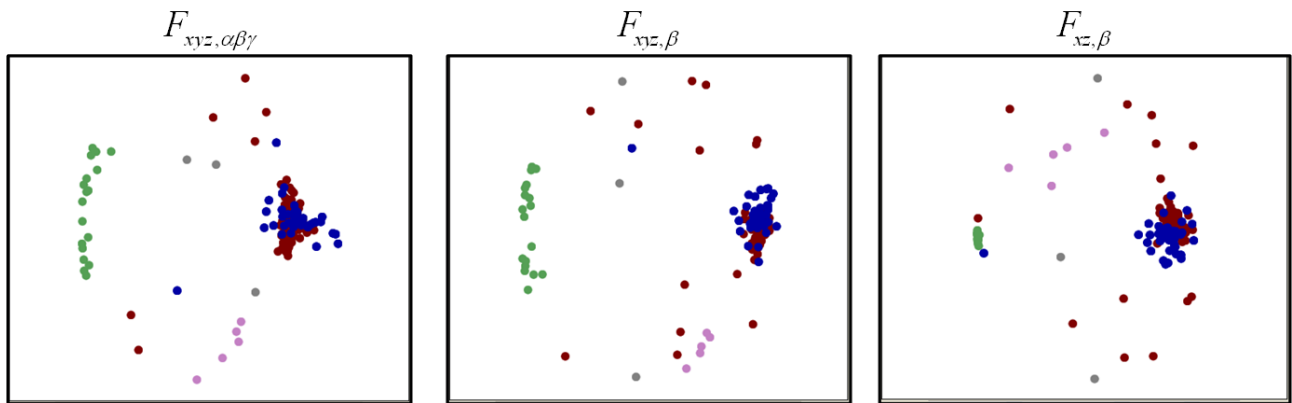


Fig. 20. The resulting scatterplot views for a velocity-base similarity metric designed to distinguish between walking (red) and running (blue) mocap sequences. MotionVis allows a user to quickly determine that this metric does not meet its design goals regardless of the motion normalization method used.



scatterplot view. As demonstrated in Section 7.2 this coupling allows for the rapid investigation of outliers.

## 8.2 Limitations and future work

Here we discuss a number of limitations which will be addressed in the near future along with some inherent limitations of the proposed environment.

The most significant omission in the current implementation is that shape encoding has not been extended to the dissimilarity matrix view. Ideally, the rows and columns of the dissimilarity matrix should indicate not only the name of the selected scatterplot points, but also the shape used to encode them. This would make associating scatterplot points with rows and columns much easier and natural. We have left this as future work due to limitations in wxWidgets, which makes this a rather time consuming feature to implement.

As discussed in Section 6.4, the colourmap in the dissimilarity matrix view is not perceptually uniform. We plan to make use of a perceptually uniform colourmap in the future and are considering if it would be advantageous to also make the colourmap segmented.

Currently, MotionVis supports a number of predefined motion normalization techniques. A more flexible approach would be to allow the user to select which positions and rotations they wish to constrain along with whether this should be applied to each frame or only once. This would allow all possible normalization methods to be explored.

MotionVis is designed to allow key aspects of a similarity metric to be quickly understood. We have demonstrated our system using Li's similarity metric along with a simple extension of this work. It is our intent to analyze other similarity metric using MotionVis in order to discover and address weaknesses in the current design.

An inherent limitation of the proposed environment is dealing with dissimilarity matrices that have an intrinsic dimensionality greater than 2. For such matrices, it will not be possible to visualize their structure using a 2-dimensional scatterplot. There are two notable options for addressing this limitation. The first is to provide alternative visualization to the scatterplot view, such as a scatterplot matrix or parallel coordinates view, that would permit the visualization of more than 2 dimensions. Another approach is to modify the mocap database being considered. Ideally, we would like to visualize the entire database in order to maximize the information available for understanding a similarity metric. However, when necessary, we could gain information about the similarity metric by considering different subsets of the mocap database that are suitable for visualizing with a single 2-dimensional scatterplot.

A similar limitation is the scalability of the proposed environment. For mocap databases with thousands instead of hundreds of mocap sequences, the proposed environment may prove to be insufficient since the scatterplot view will quickly become challenging to understand. This could be addressed by considering only a small subsets of the database at any one time. This is reasonable, but care must be taken to select subsets that will reveal useful structural information.

Alternatively, the scatterplot view could be extended to support interactive filtering in order to reduce the complexity of the visualization.

## 8.3 SimilarityVis: analyzing similarity metrics

Given the prevalence of similarity metrics used in the natural and applied sciences, a general environment for analyzing similarity metrics would be of great value. MotionVis demonstrates that a well-designed visualization environment does allow key aspects of a similarity metric to be determined. We believe the general framework of MotionVis, as illustrated in Figure 1, is suitable for any data type. That is, the structure of a database under a given similarity metric can be visualized using a scatterplot created by applying MDS to a dissimilarity matrix. Key aspects of the similarity metric can then be discovered by investigating the global and local structures imposed on the database by the similarity metric.

The following major issues would need to be addressed in order to create a generalized environment for analyzing similarity metrics:

- *Adaptive scatterplot view*: the scatterplot view should automatically reflect the database being considered. Specifically, it must encode the different classes within the data using a suitable visual variable. This encoding should allow the global structure between different classes and subclasses to be easily understood. It should also contain a notion of guaranteed visibility and would benefit from providing different means to filter the data.
- *Customized details-on-demand views*: details-on-demand views that are customized to the data type are necessary for rapidly understanding the local structure in a scatterplot view. An architecture would need to be established that allows custom views to be easily incorporated into the visualization environment. Ideally, these views should be constructed by individuals with an understanding of InfoVis principles.
- *Plug-in architecture for similarity metrics*: an architecture should be established that allows different similarity metrics to be considered without requiring the user to have more than a superficial understanding of the visualization environment's codebase. In contrast, creating custom details-on-demand views would require an understanding of the API provided by the visualization environment.

## 9 CONCLUSIONS

A wide range of similarity metrics have been developed for comparing mocap sequences. This paper specifies the design of an interactive environment for analyzing mocap-based similarity metrics. We justify the design of our environment by examining a number of scenarios of use and by relating our design decisions to well-established InfoVis practices. By tightly coupling a scatterplot view that indicates global structure with a number of details-on-demand views, our environment allows for the rapid understanding of the structure imposed on a mocap database by a given similarity metric. Understanding this structure provides the user with key insights into their similarity metric that are difficult to obtain using existing visualization environments or by analyzing numerical results.

## ACKNOWLEDGMENTS

The author would like to thank Tamara Muzner for her insightful comments related to this project. This work was partially supported by a research grant from the Natural Science and Engineering Research Council of Canada. The data used in this project was obtained from mocap.cs.cmu.edu and created with funding from NSF EIA-0196217.

## REFERENCES

- [1] Y. Sakamoto, S. Kuriyama, and T. Kaneko, "Motion map: image-based retrieval and segmentation of motion data," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on computer animation*, 2004.
- [2] E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, and M. Cardle, "Indexing large human-motion databases," in *Proceedings of the Thirtieth international conference on Very large data bases*, 2004.
- [3] M. Müller, T. Röder, and M. Clausen, "Efficient content-based retrieval of motion capture data," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 677–685, 2005.
- [4] R. Heck and M. Gleicher, "Parametric motion graphs," in *Proceedings of Symposium on Interactive 3D Graphics and Games*, 2007.
- [5] A. Safonova and J. K. Hodgins, "Construction and optimal search of interpolated motion graphs," in *ACM Transactions on Graphics (SIGGRAPH 2007)*, 2007.
- [6] L. Kovar and M. Gleicher, "Automated extraction and parameterization of motions in large data sets," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 559–568, 2004.
- [7] C. Li and B. Prabhakaran, "Indexing of motion capture data for efficient and fast similarity search," *Journal of Computers*, vol. 1, no. 3, pp. 35–42, 2006.
- [8] G. Liu, J. Zhang, W. Wang, and L. McMillan, "A system for analyzing and indexing human-motion databases," in *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM Press, 2005, pp. 924–926.
- [9] C. Ware, *Information visualization: perception for design (2nd Ed.)*. Morgan Kaufmann Publishers Inc., 2004.
- [10] T. Röder, "Similarity, retrieval, and classification of motion capture data," Ph.D. dissertation, Bonn University, 2006.
- [11] G. Johansson, "Visual motion perception," *Scientific American*, vol. 232, pp. 76–89, 1975.
- [12] J. K. Hodgins, J. F. O'Brien, and J. Tumblin, "Perception of human motion with different geometric models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 4, pp. 307–316, 1998.
- [13] A. MacEachren, X. Dai, F. Hardisty, D. Guo, and G. Lengeric, "Exploring high-d spaces with multiform matrices and small multiples," in *Proc InfoVis*, 2003.
- [14] E. J. Wegman, "Hyperdimensional data analysis using parallel coordinates," *Journal of the American Statistical Association*, vol. 85, pp. 664–675, 1990.
- [15] M. Ward, "Xmdvtool: Intergrating multiple methods for visualizing multivariate data," in *Proc. Visualization*, 1994.
- [16] A. Buja, D. Cook, and D. Swayne, "Interactive high-dimensional data visualization," *Journal of Computational and Graphical Statistics*, vol. 5, pp. 78–99, 1996.
- [17] S. Kaski, "Data exploration using self-organizing maps," *Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series*, vol. 82, p. 57, 1997.
- [18] R. DeJordy, S. P. Borgatti, C. Roussin, and D. S. Halgin, "Visualizing proximity data," *Field Methods*, vol. 19, no. 3, pp. 239–263, 2007.
- [19] W. S. Torgerson, *Theory and methods of scaling*. New York: John Wiley, 1958.
- [20] A. Buja, D. F. Swayne, M. L. Littman, and N. Dean, "Xgvis: Interactive data visualization with multidimensional scaling," [www.research.att.com/areas/stat/xgobi/papers/xgvis.pdf](http://www.research.att.com/areas/stat/xgobi/papers/xgvis.pdf), [accessed December 5, 2007].
- [21] J. Mackinlay, "Automating the design of graphical presentations of relational information," *ACM Trans. Graph.*, vol. 5, pp. 110–141, 1986.
- [22] B. Berline and P. Kay, *Basic Color Terms: Their Univesality and Evolution*. University of California Press, Berkeley, 1969.
- [23] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, pp. 1–27, 1964.

- [24] F. W. Young, *Multidimensional Scaling, Encyclopedia of Statistical Sciences (Volume 5)*, Kotz-Johnson, Ed. John Wiley & Sons, Inc., 1985.
- [25] J. C. Roberts, "State of the art: coordinated and multiple views in exploratory visualization," in *Conference on Coordinated & Multiple Views in Exploratory Visualization*, 2007.
- [26] E. Tufte, *Envisioning Information*. Graphics Press, 1990.

## APPENDIX A: COLOUR BLINDNESS EVALUATION

Approximately 5% of the population suffers from some form of colour blindness. We have designed MotionVis to be usable by these individuals. The MotionVis interface as perceived by a person with deuteranope or protanope colour blindness are given in Figures 21 and 22, respectively. These images were created using VisCheck<sup>10</sup>.

10. [www.vischeck.com](http://www.vischeck.com)

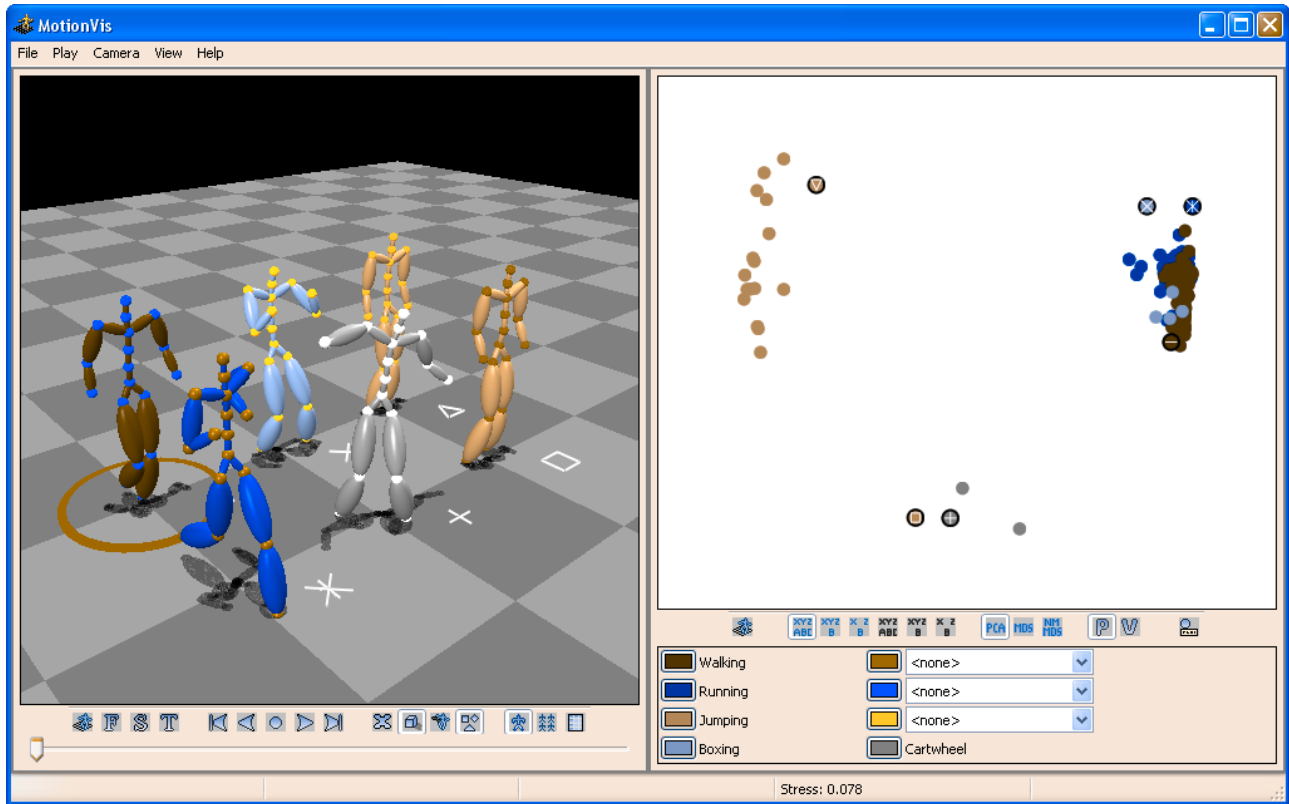


Fig. 21. MotionVis interface as perceived by a person with deuteranope colour blindness.

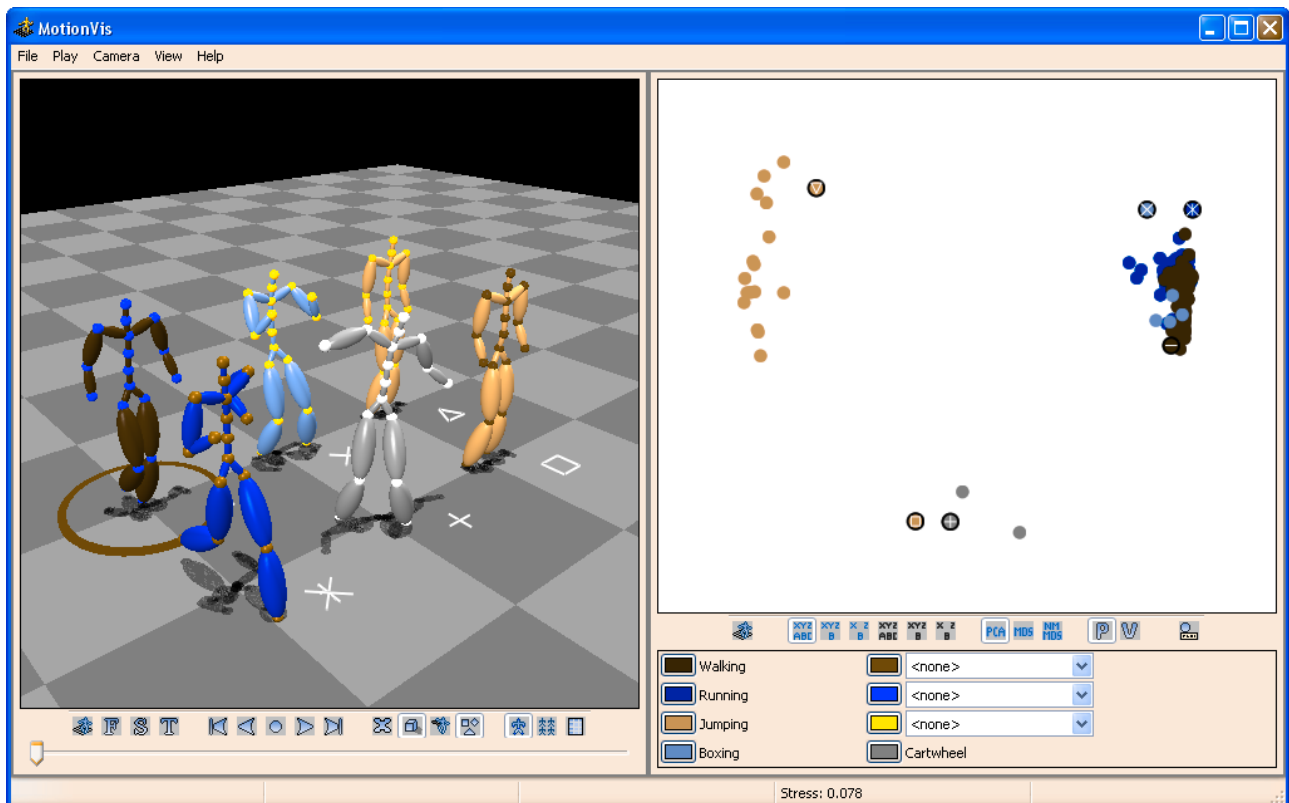


Fig. 22. MotionVis interface as perceived by a person with protanope colour blindness.